Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2419883 A1 2004/08/26

(21) 2 419 883

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

10/718,009 PTO-892

(22) Date de dépôt/Filing Date: 2003/02/26

(41) Mise à la disp. pub./Open to Public Insp.: 2004/08/26

(51) Cl.Int.^{7}/Int.Cl.^{7} G06F 17/30

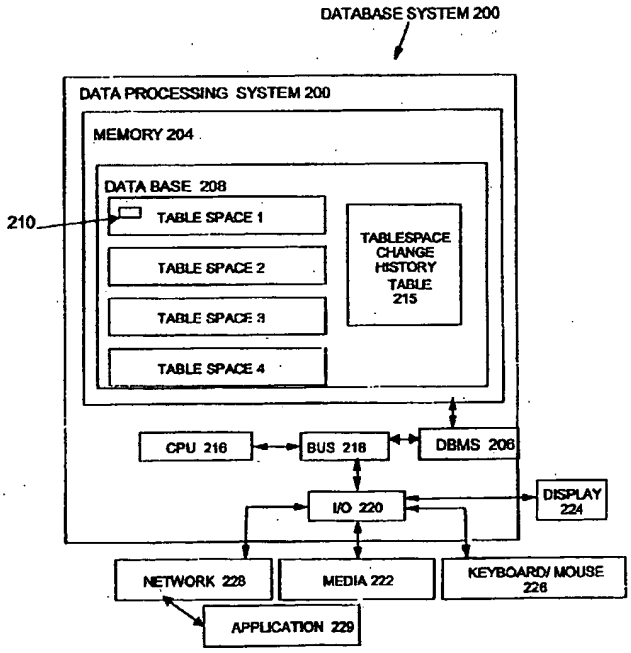
(71) Demandeur/Applicant:
IBM CANADA LIMITED - IBM CANADA LIMITEE, CA

(72) Inventeurs/Inventors:
OFER, EFFI, CA;
HURAS, MATTHEW A., CA;
WINER, MICHAEL J., CA;
ZHENG, ROGER L. Q., CA;
MCINNIS, DALE M., CA

(74) Agent: ROSEN, ARNOLD

(54) Titre : LECTURE DISCRIMINATOIRE DE FICHIERS JOURNAUX DURANT LA RECUPERATION D'ESPACE DE TABLES DANS UN SYSTEME DE GESTION DE BASE DE DONNEES

(54) Title: DISCRIMINATORY REPLAY OF LOG FILES DURING TABLE SPACE RECOVERY IN A DATABASE MANAGEMENT SYSTEM



(57) Abrégé/Abstract:
A system and method for selectively processing log files for enhancing performance of table space recovery by processing only those log files required, as well as choosing which log file to process during recovery for both redo and rollback phases of recovery. The system and method skips the processing of log files that do not contain records of interest for the table space being recovered and determines whether a log file contains anything that needs to be played, in particular for the recovery of a subsystem in the database, such as a table space. Pre processing of the log files is not done, rather including tablespace ID correlated with log file ID information is collected while the log files are created. The cross-correlated information is used during tablespace recovery to selectively determine which of the log files to process. Some log files may be skipped in cases where not all of the database system is being recovered. A sub set of the database can be recovered, such as a tablespace, and which log files to process is determined based on lock intent.

DISCRIMINATORY REPLAY OF LOG FILES DURING TABLE SPACE RECOVERY IN A DATABASE MANAGEMENT SYSTEM

ABSTRACT

A system and method for selectively processing log files for enhancing performance of table space recovery by processing only those log files required, as well as choosing which log file to process during recovery for both redo and rollback phases of recovery. The system and method skips the processing of log files that do not contain records of interest for the table space being recovered and determines whether a log file contains anything that needs to be played, in particular for the recovery of a subsystem in the database, such as a table space. Pre processing of the log files is not done, rather including tablespace ID correlated with log file ID information is collected while the log files are created. The cross-correlated information is used during tablespace recovery to selectively determine which of the log files to process. Some log files may be skipped in cases where not all of the database system is being recovered. A sub set of the database can be recovered, such as a tablespace, and which log files to process is determined based on lock intent.

DISCRIMINATORY REPLAY OF LOG FILES DURING TABLE SPACE RECOVERY IN A DATABASE MANAGEMENT SYSTEM

Field of the Invention

The present invention relates to database management systems. More specifically, the present invention relates to discriminatory replay of log files during table space recovery in a database management system.

Background

A database management system (DBMS) is a software system that facilitates the creation, maintenance, and use of an electronic database. The software system is a suite of programs which typically manage large structured sets of persistent data, offering ad hoc query facilities to many users. The DBMS controls the organization, storage and retrieval of data (fields, records and files) in the database. The DBMS also controls the security and integrity of the database. The DBMS accepts requests for data from an application program and instructs the operating system to transfer appropriate data as requested. When the DBMS is used, information systems can be changed much more easily as the organization's information requirements change. New categories of data can be added to the database without disruption to the existing system. Data security can prevent unauthorized users from viewing or updating the database. Using passwords, users are allowed access to the entire database or a series of database subsets, called sub-schemas or tablespaces. For example, an employee database can contain all the data about an individual employee, but one group of users may be authorized to view only payroll data, while others are allowed access to only work history and medical data of the employee database. The DBMS can maintain the integrity of the database through locks by not allowing more than one user to update the same record at the same time. The DBMS can keep duplicate records out of the database; for example, no two customers with the same customer numbers (key fields) can be entered into the database.

Query languages and report writers allow users to interactively interrogate the database and analyze its data. If the DBMS provides a way to interactively enter and update the database, as well as interrogate it, this capability allows for managing personal databases. However, it may

not leave an audit trail of actions or provide the kinds of controls necessary in a multi-user organization. These controls may only be available when a set of the application programs are customized for each data entry and updating function. For example, a business information system can be made up of subjects (customers, employees, vendors, etc.) and activities (orders, payments, purchases, etc.). Database design is the process of deciding how to organize this data into record types and how the record types will relate to each other. The DBMS should mirror the organization's data structure and process transactions efficiently. Organizations may use one kind of DBMS for daily transaction processing and then move the detail onto another computer that uses another DBMS better suited for random inquiries and analysis. Overall system design decisions can be performed by data administrators and systems analysts. Detailed database design can be performed by database administrators. The three most common organizations are the hierarchical databases, network databases, and relational databases. A database management system may provide one, two or all three methods. Inverted lists and other methods can also be used. The most suitable database structure can depend on the application, on the transaction rate, and the number of inquiries made.

Known DBMSs may organize multiple tablespaces and store tables of the database. To recover selected tablespaces in the event of a system crash, a backup image of the database or the table space is restored, followed by rolling forward through the log files that were created since the backup was taken. Log files contain log records which describe the changes made to the data currently stored in the database. Each log file contains one or more log records that apply to one or more tablespaces. Current recovery protocols either process or preprocess each log file during an operation for recovering the table space. However, one disadvantage of these protocols is that only those log records that apply to the tablespace being recovered need be processed. Therefore, processing all potential log files can result in inefficiencies concerning log file access and use. For example, if there was only one transaction that affected the table space being recovered, and that existed in the life span of only one log file, all the log files will still be processed. It will be appreciated that much time can be wasted in the current recovery protocols, since regardless of whether the log file contains transactions that are relevant for the table space being recovered, that log file will be processed as part of the recovery as long as it was created between the start of the backup being recovered and the point in time to which the recovery is made.

For example, referring to European Patent 2002/0007363 A1, a system and method is described for processing through all log files but filtering the ones it actually plays. This system has to go through all the log files in order to pick specific objects to recover. This system can be inefficient and inconvenient because processing time can be wasted when the system cannot skip the processing of log files that do not contain records of interest for the table space being recovered.

Referring to US Patent 6,185,577 B1, a system and a method is described for determining if a rollback record has already been played. However, this system does not determine if the record needs to be played but takes for granted that it does. A function is described for storing multiple actions to be played within a single log record. Disadvantageously, this system can not selectively process log files, which can result in processing time being wasted on correlation operations, Furthermore, the system cannot ascertain whether the log file contains anything that needs to be played.

Referring to US Patent 6,182,241, a method is described for recovering a system that terminated unexpectedly. The recovery operation includes partial processing and postponing the full processing of some non-terminated transactions to a later stage. One disadvantage is that all non-terminated transactions and therefore log records have to be processed sooner or later. Inconveniently, there is no way to skip processing of any log files or log records of non-terminated transactions. This system can also be inconvenient because it does not recover the subsystems in the database (i.e. tablespaces).

Referring to US Patent 6,178,427, a system and a method is described for dealing with mirroring log files and then extracting relevant log records from the log files so that only the table spaces being recovered are processed. However, the log files require processing prior to actual recovery in order to make it possible to skip log records by determining those specific files that may not be needed. This system can be inconvenient because it requires preprocessing of the log file.

Referring to US Patent 6,052,695, a recovery mechanism for a distributed system is described. All the log files that contain transactions after the failure must be processed. This arrangement can be inconvenient because irrelevant files are not skipped, which causes additional processing time.

Accordingly, a solution that addresses, at least in part, this and other shortcomings is desired.

Summary

It is an object of the present invention to provide a system, computer program product, network downloadable code, and a method to selectively replay log files for database recovery.

The present invention provides a method and a system for discriminatory replay of log files during table space recovery in a database management system, by identifying which log files to process during the recovery. Log files are read and processed only if they contain log records relevant to the tablespace being recovered. To know which log files are required for a given table space recovery, information cross correlating the log files with the tablespaces modified is maintained during run time. The information contains the list of table spaces that are affected by each log file. This correlation information is written to a flat file (although could be written into the database itself). During recovery, before processing a given log file, the correlation information collected during the run time is checked to verify that the selected log file is really needed for the recovery. To identify which table spaces are affected by a given log file, each transaction maintains correlation information on table spaces that it modifies. The transaction level correlation information is collected after determining the intent to change tablespaces based on lock intent. When a transaction terminates (commit or aborts), the correlation information is collected to an aggregation process. This process aggregates the correlation information sent to it from all the transactions. Therefore, when a selected log file becomes inactive, all the transactions which wrote log records in the life span of the log file have their correlation information uploaded to the aggregate process. At this point, the process writes out the correlation information for the log file that became inactive.

According to the present invention there is provided, for a database management system having a database, a tablespace contained in the database, a backup version of the tablespace contained in the database, and a log file representing changes made to the tablespace as a result of transaction executed against the tablespace subsequent to the making of the backup version, a method for directing the database management system to recover a selected tablespace, the method including the steps of: monitoring an executing transaction having an intention to modify data stored in the tablespace based on lock intent of the transactions, the modified data

represented by the contents of the log file having a log file identifier, collecting correlation information related to the modified data, the correlation information including a tablespace identifier of the modified tablespace correlated with the log file identifier, aggregating the correlation information related to the modified data, and selectively using the aggregated correlation information to discriminately execute selectable transactions logged in the log file against the backup version of the tablespace by matching the log file identifier with the tablespace identifier.

According to a further aspect of the present invention there is provided, for a database management system having a database, a tablespace contained in the database, a backup version of the tablespace contained in the database, and a log file representing changes made to the tablespace as a result of a transaction executed against the tablespace subsequent to the making of the backup version, a computer program product having a computer-readable medium tangibly embodying computer executable instructions for directing a database management system to recover a selected tablespace, the computer program product including: computer readable code for monitoring an executing transaction having an intention to modify data stored in the tablespace based on lock intent of the transactions, the modified data represented by the contents of the log file having a log file identifier, computer readable code for collecting correlation information related to the modified data, the correlation information including a tablespace identifier of the modified tablespace correlated with the log file identifier, computer readable code for aggregating the correlation information related to the modified data, and computer readable code for selectively using the aggregated correlation information to discriminately execute selectable transactions logged in the log file against the backup version of the tablespace by matching the log file identifier with the tablespace identifier.

According to a further aspect of the present invention there is provided, for a database management system having a database, a tablespace contained in the database, a backup version of the tablespace contained in the database, and a log file representing changes made to the tablespace as a result of a transaction executed against the tablespace subsequent to the making of the backup version, an article including a computer-readable signal-bearing medium usable on a network, and also including means in the medium for directing a database management system to recover a selected tablespace, the article including: means in the medium for monitoring an executing transaction having an intention to modify data stored in the tablespace based on lock

intent of the transactions, the modified data represented by the contents of the log file having a log file identifier, means in the medium for collecting correlation information related to the modified data, the correlation information including a tablespace identifier of the modified tablespace correlated with the log file identifier, means in the medium for aggregating the correlation information related to the modified data, and means in the medium for selectively using the aggregated correlation information to discriminately execute selectable transactions logged in the log file against the backup version of the tablespace by matching the log file identifier with the tablespace identifier.

According to a further aspect of the present invention there is provided a database management system having a database, a tablespace contained in the database, a backup version of the tablespace contained in the database, and a log file representing changes made to the tablespace as a result of a transaction executed against the tablespace subsequent to the making of the backup version, the database management system adapted to recover a selected tablespace, the database management system including: a transaction module for monitoring an executing transaction having an intention to modify data stored in the tablespace based on lock intent of the transactions, the modified data represented by the contents of the log file having a log file identifier, the transaction module collecting correlation information related to the modified data, the correlation information including a tablespace identifier of the modified tablespace correlated with the log file identifier, an aggregation module for aggregating the correlation information related to the modified data, and a recovery module for selectively using the aggregated correlation information to discriminately execute selectable transactions logged in the log file contents against the backup version of the tablespace by matching the log file identifier with the tablespace identifier.

A better understanding of these and other embodiments of the present invention can be obtained with reference to the following drawings and description of the preferred embodiments.

Brief Description of the Drawings

The embodiments of the present invention will be explained by way of the following drawings, in which:

Figure 1 shows an operation for recovering a table space to a point in time;

Figure 2 shows a database system for implementing tablespace recovery;

Figure 3 shows a timeline of log files and transactions within the database system of Figure 2;

Figure 4 shows a table space change history file of a database management system of Figure 2;

Figure 5 shows modular components of the database management system of Figure 2;

Figure 6 shows operation of a transaction code module of Figure 5;

Figure 7 shows operation of an aggregator code module of Figure 5; and

Figure 8 shows operation of a table space recovery module of Figure 5.

It should be noted that similar references are used in different figures to denote similar components.

Detailed Description

The following detailed description of the embodiments of the present invention does not limit the implementation of the invention to any particular computer programming language. The present invention may be implemented in any computer programming language provided that the OS (Operating System) provides the facilities that may support the requirements of the present invention. A preferred embodiment is implemented in the C or C++ computer programming language (or other computer programming languages in conjunction with C/C++). Any limitations presented would be a result of a particular type of operating system or computer programming language and would not be a limitation of the present invention.

Embodiments of the present invention provide a method, a data processing system, a computer program product, and/or an article for directing a database management system to recover a selected tablespace being contained in a database, the database being adapted to contain a backup version of the tablespace and a log file having contents being adapted to represent changes made to the tablespace subsequent to the backup version by at least one transaction executed by the database management system against the tablespace.

Other embodiments of the present invention provide a method, a data processing system, a computer program product, and/or an article for implementing a data structure usable in a

recovery by a database management system of selected tablespaces contained in a database, the database being adapted to contain a backup version of the tablespaces and a plurality of log files having respective contents being adapted to represent changes made to the tablespaces subsequent to the backup version by at least one transaction executed by the database management system against the tablespaces, the transactions having an intent to modify the tablespaces based on lock intent.

It will be appreciated, by those skilled in the art, that the article can be a signal bearing medium for transporting computer readable code to a data processing system over a network, in which the code can be used to implement the method. It will also be appreciated, by those skilled in the art, that the computer program product includes a computer readable medium having computer executable code for directing a data processing system to implement the method. The computer program product can also be called a computer-readable memory, in which the memory can be a CD, floppy disk or hard drive or any sort of memory device usable by a data processing system. It will also be appreciated, by those skilled in the art, that a data processing system may be configured to operate the method (either by use of computer executable code residing in a medium or by use of dedicated hardware modules, also generally or generically known as mechanisms or means, which may operate in an equivalent manner to the code which is well known in the art).

Referring to Figure 1, an example recovery operation 100 is shown for recovering a table space of a database (see Figure 2) to a point in time 101. For example, sometime in between Tuesday and Wednesday a database administrator detects that an error 102 has occurred within the table space. A database management system (see Figure 2) is used to recover the table space with minimal errors by restoring a backup version of the table space (indicated as backup 104), from Monday. The database management system obtains the backup version 104 of the table space and begins a roll forward operation 106 of selected log files 107 to the beginning of Tuesday. It is noted that each log file 107 can contain many log records. Each log record records a transaction that interacted with the various tablespaces contained in the database. Typically, the roll forward operation can include processing selected log files in a serial manner (such as starting from log file #10 and onwards to log file #14 in a discriminatory manner as further described below).

Referring to Figure 2, a database system 200 is implemented in a data processing system 202 having memory 204 coupled to a bus 218. Coupled to bus 218 are other components, such as a CPU (Central Processing Unit) 216 and I/O subsystem 220. CPU 216 executes instructions stored in memory 204, such as a DBMS (Database Management System) 206. Operatively coupled to I/O subsystem 220 is a network 228, a media 222, a keyboard/mouse 226, and a display unit 224, all known in the art. The media 222 may include code (such as the DBMS 206) which may be transferred for resident storage in the memory 204 via an I/O subsystem 220 and the bus 218. Also stored in the memory 204 is a database 208 including a collection of table spaces 1,2,3,4, and a table space change history information table 215. Further, application programs 229 can interact with the database 208 over the network 228. The application programs 229 request data objects 210 and their modification during interaction with the database 208. It is noted that the tablespaces 1,2,3,4 can include stored object data 210 as organized and managed by the DBMS 206. The interaction of the DBMS 206 and table space change history table 215 will be described below in greater detail.

It will be appreciated that database system 202 may be stored in the memory of data processing system 200 or stored in a distributed data processing system (not depicted). Data processing system 200 includes the CPU 216 (Central Processing Unit) operatively coupled to memory 204, which also stores an operating system (not depicted) for general management of the data processing system 200. An example of the data processing system 200 is such as but not limited to the IBM™ ThinkPad™ computer. The database system 200 includes computer executable programmed instructions for directing the data processing system 202 to implement embodiments of the methods and data processing systems having means for implementing those methods. The programmed instructions can be embodied on a computer readable medium (such as but not limited to a CD disk or floppy disk) which may be used for transporting the programmed instructions to the memory 204 of data processing system 202. Alternatively, the programmed instructions may be embedded in a computer-readable, signal-bearing medium that is uploaded to the network 228 by a vendor or supplier of the programmed instructions, and this signal-bearing medium may be downloaded to the data processing system 202 from the network 228 by such as but not limited to end users or potential buyers.

It will be appreciated that an desirable aspect of the invention may be provided by a computer program product having a computer-readable medium tangibly embodying computer

executable instructions for directing the data processing system 202 to implement any method or data processing system 202 to be described below. It will be appreciated that the computer program product may be a floppy disk, hard disk or other medium for long term storage of the computer executable instructions.

It will be appreciated that an aspect of the invention may be provided by an article having a computer-readable signal-bearing medium, and having means in the medium for directing the data processing system 202 to implement any method to be described below. It will be appreciated that a supplier of the embodiment of the invention may upload the article to the network 228 (such as the Internet) and users may download the article via the network 228 to their respective data processing systems 202.

Figure 3 shows example timelines of log files 303 and transactions 305 realized within the database system 200 of Figure 2. Log files 303 contain log records 307 which describe the changes made to the data objects 210 currently stored in the database 208. Each log file 303 contains one or more log records 307 that apply to one or more table spaces 1,2,3,4. During an operation for recovering selected table spaces 1,2,3,4, the log files 303 are selectively read, as further described below, with those records 307 processed that apply to the table space 1,2,3,4 being recovered.

Referring again to Figure 3, time line 302 shows the log files 303 which collect tracking information (that is, information about changes made to various tablespaces 1,2,3,4 by various transactions 305). For example, log file #10 is opened and begins collecting tracking information into its set of log records 307. Each log record 307 includes the tracking details of an operation or transaction 305, which affects one or more tablespaces 1,2,3,4. For example, once log file #10 is filled, log file #11 is opened and begins collecting additional transaction information. The transaction information collection process repeats for remaining log files 303, such as log files #12, #13, and #14.

Time line 304 shows the start and end of Transaction #1. It should be noted that the actions performed by Transaction #1 are recorded in the log records 307 associated with log file #10 and log file #11. By way of example, Transaction #1 modifies tablespaces 1, 2, and 3. Similarly, time line 306 shows the start and end of Transaction #2. The actions performed by Transaction #2 are recorded in the log records 307 associated with log file #10, log file #11, and

log file #12. By way of example, Transaction #2 modifies tablespaces 2, 3, and 4. Time line 308 shows the start and end of Transaction #3. The actions performed by Transaction #3 are recorded in the log records 307 associated with log file #11, log file #12, and log file #13. By way of example, Transaction #3 modifies tablespaces 2 and 4.

It is noted that for the transaction information collected in the log files 303, it is important to ensure the ACID (Atomicity, Consistency, Isolation, and Durability) properties of the database 208. This transaction information is useful in the case where the transaction 305 is to be rolled back as well as for reply during database 208 and tablespace 1,2,3,4 recovery after a restore or a system 200 crash. Concurrency control and locking is the mechanism used by the DBMS 206 for the sharing of data objects 210. Atomicity, consistency, and isolation are achieved through concurrency control and locking, when many people may be reading the same data object 210 at the same time from the database 208. It is usually necessary to ensure that only one application 229 at a time can change selected data objects 210. Locking is a way to do this, because of locking, all changes to the particular data object 210 will be made in the correct order during concurrent transactions 305. For example, the amount of data objects 210 that can be locked with a single instance (transaction 305) or groups of instances (transactions 305) defines the granularity of the lock. In general , the types of granularity can be such as but not limited to page locking, cluster locking, class or table locking, and object or instance (transaction 305) locking.

Referring to Figure 4, the table space change history file or table 215 is contained within the database system 200 of Figure 2. Table 215 is a data structure for recording the tablespaces 1,2,3,4 that are modified by the log records 307 in specific log files 303. The table 215 contains history records 400 that can be composed of three fields, namely 402, 404, 406. The log file indicator (field 404) contains the specific ID of the log file 303 for which the record 400 applies. The table space modified (field 406) contains the list of tablespace IDs that are modified by the log records 307 in the log file 303 indicated by the log file indicator (field 404). The complete indicator (field 402) is used to record whether the information collected for the log file 303 referenced in the log file indicator field 404 is complete. The correlated information of the table 215 contains the IDs of specific tablespaces 1,2,3,4 and the associated log file IDs 407 of the specific log files 303 used to record the tablespace 1,2,3,4 modifications by the log records 307.

In the example shown in Figures 3 and 4, the specific log file #10 contains log records 307 which modify tablespaces 1,2,3 and 4. This corresponds to the information shown in Figure 3, where transaction #1 and transaction #2 are active during the life span of log file #10. As described above, transaction #1 modified tablespaces 1,2 and 3, while transaction #2 modified table spaces 2,3, and 4. Together these two transactions (#1 and #2) modified tablespaces 1,2,3 and 4.

Referring again to Figures 3 and 4, the specific log file #11 contains log records 307 which modify tablespaces 1,2,3 and 4. This corresponds to the information shown in Figure 3, where transactions #1, #2 and #3 were active during the life span of log file #11. Accordingly: transaction #1 modified tablespaces 1,2 and 3; Transaction #2 modified tablespaces 2,3, and 4; and Transaction #3 modified tablespaces 2 and 4. Together, these three transactions modified tablespaces 1,2,3 and 4.

Similarly, Figures 3 and 4 show the specific log file #12 to contain log records 307 which modify tablespaces 2,3 and 4. This corresponds to the information shown in Figure 3, where transactions #2 and #3 were active during the life span of log file 12. It should be noted that transaction #1 was no longer active during the life span of log file #12 because it was finished during log file #11. Accordingly: transaction #2 modified tablespaces 2,3, and 4; and transaction #3 modified table spaces 2 and 4. Together these two transactions modified tablespaces 2,3 and 4.

Finally, Figures 3 and 4 show the specific log file #13 containing log records 307 which modify tablespaces 2 and 4. This corresponds to the information shown in Figure 3, where only transaction #3 was active during the life span of log file #13. It should be noted that transaction #2 was no longer active during the life span of log file #13 because it was finished during log file #12. Accordingly, transaction #3 modified tablespaces 2 and 4. Therefore only tablespaces 2 and 4 are shown to be modified in the life span of log file #13.

Referring again to Figure 4, the purpose of the complete indicator 402 is to indicate that when the correlation information in the table space change history table 215 is used during recovery, only those records 400 that corresponds to log files 303 for which all the tracking information (i.e. object 210 modification information) had been collected are utilized. In an alternative embodiment, the Complete Indicator 402 may be removed because the record 400 for

a given file 303 is written to the table space change history table 215 only when all the information for that given file 303 has been collected and stored in its log records 307.

Figure 5 shows a transaction code module 502, an aggregator code module 504, and a table space recovery module 506 included with the DBMS 206 of Figure 2. Operation of the transaction code module 502 is described in greater detail in operation S600 of Figure 6. Operation of the aggregator code module 504 is described in greater detail in operation S700 of Figure 7. Operation of the recovery code module 506 is described in greater detail in operation S800 of Figure 8. Generally, the transaction code module 502 will interact with the selected tablespaces 1,2,3,4 as they are modified, and then provide table space change history information to the aggregator code module 504, in which that correlation information will be eventually stored in the table 215. Aggregator code module 504 receives the correlation information, and then transfers this information to the table space change history table 215. For example, when a database administrator needs to restore specific tablespaces 1,2,3,4 in the event of a system 200 crash, the DBMS 206 executes table space recovery module 506 to use the history records 400 of the table 215. Therefore, the transaction module 502 and the aggregation module 504 of the DBMS 206 are used to collect and update the history records 400 of the table 215, as the selected tablespaces 1,2,3,4 are modified. When desired, the DBMS 206 uses the recovery module 506 to process the specific log files 303 listed in the table 215. The log records 307 of the selected log files 303, identified by their IDs, are used to restore the correlated tablespaces 1,2,3,4 by the DBMS 206.

Figure 6 shows operation S600 of the transaction code module 502 of Figure 5. Operation S600 includes a life cycle of the transaction 305 adapted for interaction with the tablespace change history table 215 of Figure 2. Note that the correlation information (including log file ID 407 with tablespace ID) regarding which tablespaces 1,2,3,4 being modified by the transaction 305 is collected when the transaction 305 expresses an interest in obtaining an update lock on the selected tablespace 1,2,3,4. It is noted that this sort of lock is always obtained prior to objects 210 of the tablespace 1,2,3,4 being modified. For example, locked objects can be identified by file and block number. Locks can be chained by both object 210 and transaction 305 to facilitate traversal during transaction 305 commit and abort functions.

Referring to Figure 6, operation S602 includes starting operation S600 and operation S604 includes obtaining a write lock. This sort of lock is obtained whenever the objects 210 in the database 208 are to be modified, also referred to as lock intent. Once the write lock is obtained, operation S606 includes identifying the modified tablespace 1,2,3,4. Identification is made by the code that obtains the lock. This sort of information is inherent in obtaining locks, since in order to obtain the lock, the identity of the object 210 being locked is known. Part of this information is which tablespace 1,2,3,4 the object 210 resides in. Operation S608 includes using the lock to insert or otherwise modify data objects 210 associated with the tablespace 1,2,3,4 selected in the database 208. It is recognized that in order to prevent multiple transactions from modifying the same object 210 at the same time, the lock is first obtained prior to modification. Once the lock is obtained, the transaction 305 can change the object 210, such as but not limited to a row, a table, or some other object as needed.

Referring again to Figure 6, operation S610 includes transferring the identity (ID) (see Figure 4) of identified tablespaces 1,2,3,4 to the aggregator code module 504 (see Figure 5). The tablespace ID is collected as part of the transaction 305 state information. The transaction 305 maintains the list of all the tablespaces 1,2,3,4 that the transaction 305 modified. This list of tablespace IDs is sent to the aggregator module 504 during the termination of the transaction 305. Operation S612 determines whether to terminate operation S600. If termination of S600 is required, control proceeds to operation S614. If termination of operation S600 is not required, control is transferred to operation S602.

Operation S614 includes a stopping operation of the transaction module 502. When the transaction 305 is stopped, all the correlation information that was collected by the transaction 305 is transferred to the aggregator code module 504 by the transaction module 502. This correlation information includes the list of tablespace IDs modified by the transaction 305, the associated log file IDs 407, as well as some other implicit information indicating when the transactions 305 started and ended (i.e. which span of the log files 303 was affected). The identity ID of the tablespaces modified (plus the rest of the used state information) can be transferred by the transaction module 502 such as but not limited to via a function call, a message queue, a remote procedure call, shared memory, or some other communication mechanism. It is noted the transaction state can be a per transaction structure, which describes the currently running transactions 305. Attributes of the transaction state can include such as but

not limited to idle, running, aborting, and committing, associated log files 3030, a pointer to the chain of locks currently held, a transaction 305 identifier, and links to other transaction states.

Once the tablespace ID and the other state information (such as the log file numbers 407) is sent to the aggregator module 504, referring to Figure 7, the operation S700 includes collecting correlation information regarding which tablespaces 1,2,3,4 are modified by which log files 303. Once all the correlation information for a given tablespace 1,2,3,4 is collected (i.e. the information from all the transactions 305 that were alive during the life span of the log files 303), the log file complete indicator 402 is set as true. Alternatively, the log file information record 400 can simply be written out to the tablespace change history table 215 only when the correlation information has been fully collected.

Referring again to Figure 7, operation S702 includes starting the operation S700, and operation S704 includes obtaining the identity ID of identified table spaces 1,2,3,4. Operation S706 includes obtaining start and end time of completed transaction 305 for correlating the log file IDs 407 of the affected log files 303. This correlation information is maintained as part of the state of the transaction 305 and is sent to the aggregator module 504 when the transaction 305 terminates. Operation S708 includes aggregating the obtained correlation information into the appropriate record 400 of the table space change history table 215. Operation S710 includes determining whether processing of all log records 307 of the current log file 303 is completed. If processing of all log records 307 of the log file 303 has been completed, control is transferred to operation S712. If processing of all log records 307 of the log file 303 has not been completed, control is transferred to operation S704. Operation S712 includes setting the complete indicator 402 for log file identifiers 407 in the table space change history table 215. Alternatively, operation S712 writes the record 400 for the corresponding log file 303 to disk (not shown) of the database 208. Operation S714 includes a stopping operation of the aggregation module 504.

Referring to Figure 8, operation S800 of table space recovery code module 506 is shown. Operation S800 includes recovering the selected tablespace 1,2,3,4 and selectively processing the correlated log files 303 indicated by the log file IDs 407 in the list under heading log file indicator 404 of the table 215. Operation S802 includes a starting operation of the recovery module 506. Operation S804 includes receiving a command from the DBMS 206 to restore the selected tablespace 1,2,3,4. Operation S806 includes starting the roll forward

operation, with operation S808 selecting the log file IDs 407 from the log file indicator 404, which are correlated with the tablespace 1,2,3,4 as listed in the tablespaces modified 406, i.e. specific log files #10 and #11 are only associated with recovering tablespace 1, while specific log files #10, #11, #12 would be needed for recovering tablespace 3. Similarly, specific log files #10, #11, #12, #13 would be needed for recovering tablespaces 2 and 4. It is recognized that the first log file ID 407 to be selected is the first log file 303 that became active after the backup started.

Operation S810 includes determining whether the selected log file 303 has its corresponding log file ID 407 listed in the tablespace 1,2,3,4 change history table 215. If yes, control is transferred to operation S812. If no, control is transferred to operation S814. Note that if there is no record 400 in the table space change history table 215 for the selected log file ID 407, that log file ID 407 can be assumed to be needed for the recovery. Operation S812 includes processing the selected log file 303, and then control is transferred back to operation S808. Operation S814 includes selecting another log file ID 407 for checking if listed in the table 215. It is noted that the next file ID 407 selected in the table 215 is the file 303 that became active after the current file 303. Operation S816 includes determining whether to end operation S818 of the recovery module 506. If operation S800 is to end, control is transferred to operation S818. If operation S800 is to continue, control is transferred to operation S810.

In an alternative embodiment of operation S800, an initial operation includes selecting Select next log file ID 407 to process. Another operation includes determining whether there is a log file ID 407 needed for recovery. If yes, the corresponding log file 303 is processed, and the next file ID 407 is set to process to the next sequential log file 303. If no, then another log file ID 407 is picked from the table 215 for processing. Both subsequent yes/no branches continue to another operation which includes determining whether there is a next log file 303 to process, corresponding to the file ID 407 from the table 215. If yes, then go back to the "is log file 303 needed for recovery". If no, then end or stop operations are implemented.

Therefore, referring to Figures 1, 2, 4, and 5, recovery of the tablespace objects 210 in the database system 200 uses the restoring of the table space backup version 104, followed by rolling forward through all correlated log files 303 (i.e. the non-listed log files 303 are skipped) containing the changes that took place after the backup 104 was recorded. The log files 303 are

used to record the changes that happen to all table spaces 1,2,3,4 in the database system 200, through the series of log records 307. However, the roll forward operation 106 only uses those records 307 in the log files 303 that relate to the particular table space 1,2,3,4 being rolled forward (i.e. recovered). The database management system 206 notes when transactions 305 obtain locks on the objects 210 within selected table spaces 1,2,3,4, with the intention to perform a modification of the tablespace 1,2,3,4. The DBMS 206 marks or otherwise correlates the transaction 305 with the associated log files 303 in the history table 215, as modifying the tablespace 1,2,3,4 in which the objects 210 reside. During the transaction 305 termination (such as commit or abort), the correlation information relating to the tablespaces 1,2,3,4 being modified is monitored and recorded by the modules 502, 504. These modules 502, 504 help to aggregate the correlation information collected from all transactions 305 of the database system 200 at the transaction level (i.e. log file 303 granularity) and this information is recorded in the history table 215, such as but not limited to a flat file. During recovery, the table 215 is processed to determine which log files 303 should be processed and which log files 303 should be skipped.

The method of selectively or discriminately replaying log files 303 by the DBMS 206 includes; determining the intent to change data of objects 210 by noting lock intent; collecting correlation information from all transactions 305 by marking each transaction 305 as modifying the tablespace 1,2,3,4 in which the object 210 resides; and aggregating the correlation information from all the transactions 305 at a log file 303 level (log files granularity); and writing the information to the history table 215. The DBMS 206 then uses the correlation information contained in the history table 215 to selectively determine which of the log files 303 should be handled (and corresponding log records 307) during recovery of selected tablespaces 1,2,3,4.

In a further embodiment, database machines can be specially designed computers for holding the actual database system 200 and run only the DBMS 206 and related software. Connected to one or more mainframes via a high-speed channel, the database machines can be used in large volume transaction processing environments. Database machines can have a large number of DBMS 206 functions built into the hardware and can also provide special techniques for accessing the disks (not shown) containing the databases 208, such as using multiple

processors concurrently for high-speed searches. The database objects 210 can be made up of data, text, pictures and voice.

It will be appreciated that variations of some elements are possible to adapt the invention for specific conditions or functions. The concepts of the present invention can be further extended to a variety of other applications that are clearly within the scope of this invention. Having thus described the present invention with respect to a preferred embodiments as implemented, it will be apparent to those skilled in the art that many modifications and enhancements are possible to the present invention without departing from the basic concepts as described in the preferred embodiment of the present invention. Therefore, what is intended to be protected by way of letters patent should be limited only by the scope of the following claims.

CLAIMS:

The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

1. For a database management system having:

a database,

a tablespace contained in the database,

a backup version of the tablespace contained in the database, and

a log file representing changes made to the tablespace as a result of a transaction executed against the tablespace subsequent to the making of the backup version,

a method for directing the database management system to recover a selected tablespace, the method comprising the steps of:

monitoring an executing transaction having an intention to modify data stored in the tablespace based on lock intent of the transactions, the modified data represented by the contents of the log file having a log file identifier;

collecting correlation information related to the modified data, the correlation information including a tablespace identifier of the modified tablespace correlated with the log file identifier;

aggregating the correlation information related to the modified data; and

selectively using the aggregated correlation information to discriminately execute selectable transactions logged in the log file against the backup version of the tablespace by matching the log file identifier with the tablespace identifier.

2. The method of claim 1 wherein the aggregated correlated information of the executed transactions is aggregated at a log file granularity.

3. The method of claim 2 further comprising the step of writing the aggregated correlation information to a transaction history table in a predefined format.

4. The method of claim 3, wherein the table is adapted to contain the correlation information having a list of at least two of the log file identifiers associated with the tablespace identifier.

5. The method of claim 4 further comprising the step of processing the list of correlated log file identifiers in the table for selecting which of the log files are associated with the tablespace.

6. The method of claim 5 further comprising the step of processing the selected log files according to the respectively selected log file identifiers for recovering the tablespace, wherein recovering the tablespace includes rolling forward through all the selected log files from the list associated with the tablespace and skipping those log files whose log file identifiers are not on the list.

7. The method of claim 3, wherein the aggregated correlation information is adapted to include multiple tablespace identifiers with correlated multiple log file identifiers.

8. The method of claim 3, wherein the aggregated correlation information includes at least two log file identifiers correlated with the tablespace identifier.

9. The method of claim 3, wherein the tablespace identifier and a second tablespace identifier are correlated with the log file identifier in the table.

10. The method of claim 4, wherein the predefined format includes a listing of the log file identifiers and a listing of associated tablespace identifiers.

11. For a database management system having:

a database,

a tablespace contained in the database,

a backup version of the tablespace contained in the database, and

a log file representing changes made to the tablespace as a result of a transaction executed against the tablespace subsequent to the making of the backup version,

a computer program product having a computer-readable medium tangibly embodying computer executable instructions for directing a database management system to recover a selected tablespace, the computer program product comprising:

computer readable code for monitoring an executing transaction having an intention to modify data stored in the tablespace based on lock intent of the transactions, the modified data represented by the contents of the log file having a log file identifier;

computer readable code for collecting correlation information related to the modified data, the correlation information including a tablespace identifier of the modified tablespace correlated with the log file identifier;

computer readable code for aggregating the correlation information related to the modified data; and

computer readable code for selectively using the aggregated correlation information to discriminately execute selectable transactions logged in the log file against the backup version of the tablespace by matching the log file identifier with the tablespace identifier.

12. The computer program product of claim 11 wherein the aggregated correlated information of the executed transactions is aggregated at a log file granularity.

13. The computer program product of claim 12 further comprising computer readable code for writing the aggregated correlation information to a transaction history table in a predefined format.

14. The computer program product of claim 13, wherein the table is adapted to contain the correlation information having a list of at least two of the log file identifiers associated with the tablespace identifier.

15. The computer program product of claim 14 further comprising computer readable code for processing the list of correlated log file identifiers in the table for selecting which of the log files are associated with the tablespace.

16. The computer program product of claim 15 further comprising computer readable code for

processing the selected log files according to the respectively selected log file identifiers for recovering the tablespace, wherein recovering the tablespace includes rolling forward through all the selected log files from the list associated with the tablespace and skipping those log files whose log file identifiers are not on the list.

17. The computer program product of claim 13, wherein the aggregated correlation information is adapted to include multiple tablespace identifiers with correlated multiple log file identifiers.

18. The computer program product of claim 13, wherein the aggregated correlation information includes at least two log file identifiers correlated with the tablespace identifier.

19. The computer program product of claim 13, wherein the tablespace identifier and a second tablespace identifier are correlated with the log file identifier in the table.

20. The computer program product of claim 14, wherein the predefined format includes a listing of the log file identifiers and a listing of associated tablespace identifiers.

21. For a database management system having:

a database,

a tablespace contained in the database,

a backup version of the tablespace contained in the database, and

a log file representing changes made to the tablespace as a result of a transaction executed against the tablespace subsequent to the making of the backup version,

an article including a computer-readable signal-bearing medium usable on a network, and also including means in the medium for directing a database management system to recover a selected tablespace, the article comprising:

means in the medium for monitoring an executing transaction having an intention to modify data stored in the tablespace based on lock intent of the transactions, the modified data represented by the contents of the log file having a log file identifier;

means in the medium for collecting correlation information related to the modified data, the correlation information including a tablespace identifier of the modified tablespace correlated with the log file identifier;

means in the medium for aggregating the correlation information related to the modified data; and

means in the medium for selectively using the aggregated correlation information to discriminately execute selectable transactions logged in the log file against the backup version of the tablespace by matching the log file identifier with the tablespace identifier.

22. The article of claim 21 wherein the aggregated correlated information of the executed transactions is aggregated at a log file granularity.

23. The article of claim 22 further comprising means in the medium for writing the aggregated correlation information to a transaction history table in a predefined format.

24. The article of claim 23, wherein the table is adapted to contain the correlation information having a list of at least two of the log file identifiers associated with the tablespace identifier.

25. The article of claim 24 further comprising means in the medium for processing the list of correlated log file identifiers in the table for selecting which of the log files are associated with the tablespace.

26. The article of claim 25 further comprising means in the medium for processing the selected log files according to the respectively selected log file identifiers for recovering the tablespace, wherein recovering the tablespace includes rolling forward through all the selected log files from the list associated with the tablespace and skipping those log files whose log file identifiers are not on the list.

27. The article of claim 23, wherein the aggregated correlation information is adapted to include multiple tablespace identifiers with correlated multiple log file identifiers.

28. The article of claim 23, wherein the aggregated correlation information includes at least two log file identifiers correlated with the tablespace identifier.

29. The article of claim 23, wherein the tablespace identifier and a second tablespace identifier are correlated with the log file identifier in the table.

30. The article of claim 24, wherein the predefined format includes a listing of the log file identifiers and a listing of associated tablespace identifiers.

31. A database management system having:

a database,

a tablespace contained in the database,

a backup version of the tablespace contained in the database, and

a log file representing changes made to the tablespace as a result of a transaction executed against the tablespace subsequent to the making of the backup version,

the database management system adapted to recover a selected tablespace, the database management system comprising:

a transaction module for monitoring an executing transaction having an intention to modify data stored in the tablespace based on lock intent of the transactions, the modified data represented by the contents of the log file having a log file identifier, the transaction module collecting correlation information related to the modified data, the correlation information including a tablespace identifier of the modified tablespace correlated with the log file identifier;

an aggregation module for aggregating the correlation information related to the modified data; and

a recovery module for selectively using the aggregated correlation information to discriminately execute selectable transactions logged in the log file contents against the backup version of the tablespace by matching the log file identifier with the tablespace identifier.

32. The database management system of claim 31, wherein the aggregated correlated information of the executed transactions is aggregated at a log file granularity.

33. The database management system of claim 32, wherein the aggregator module further comprises writing the aggregated correlation information to a transaction history table in a predefined format.

34. The database management system of claim 33, wherein the table is adapted to contain the correlation information having a list of at least two of the log file identifiers associated with the tablespace identifier.

35. The database management system of claim 34, wherein the recovery module further comprises processing the list of correlated log file identifiers in the table for selecting which of the log files are associated with the tablespace.

36. The database management system of claim 35, wherein recovering the tablespace includes processing the selected log files according to the respectively selected log file identifiers and rolling forward through all the selected log files from the list associated with the tablespace and skipping those log files whose log file identifiers are not on the list.

37. The database management system of claim 33, wherein the aggregated correlation information is adapted to include multiple tablespace identifiers with correlated multiple log file identifiers.

38. The database management system of claim 33, wherein the aggregated correlation information includes at least two log file identifiers correlated with the tablespace identifier.

39. The database management system of claim 33, wherein the tablespace identifier and a second tablespace identifier are correlated with the log file identifier in the table.

40. The database management system of claim 34, wherein the predefined format includes a listing of the log file identifiers and a listing of associated tablespace identifiers.

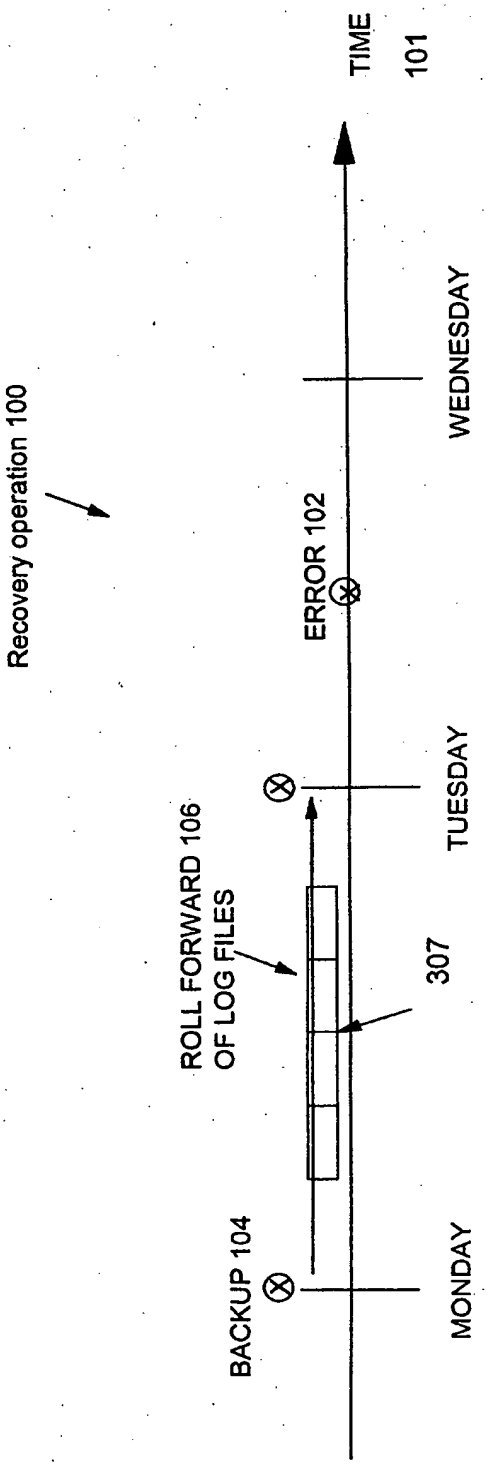

Recovery operation 100
TIME
101
ERROR 102
WEDNESDAY
TUESDAY
MONDAY
ROLL FORWARD 106
OF LOG FILES
307
BACKUP 104

FIG. 1

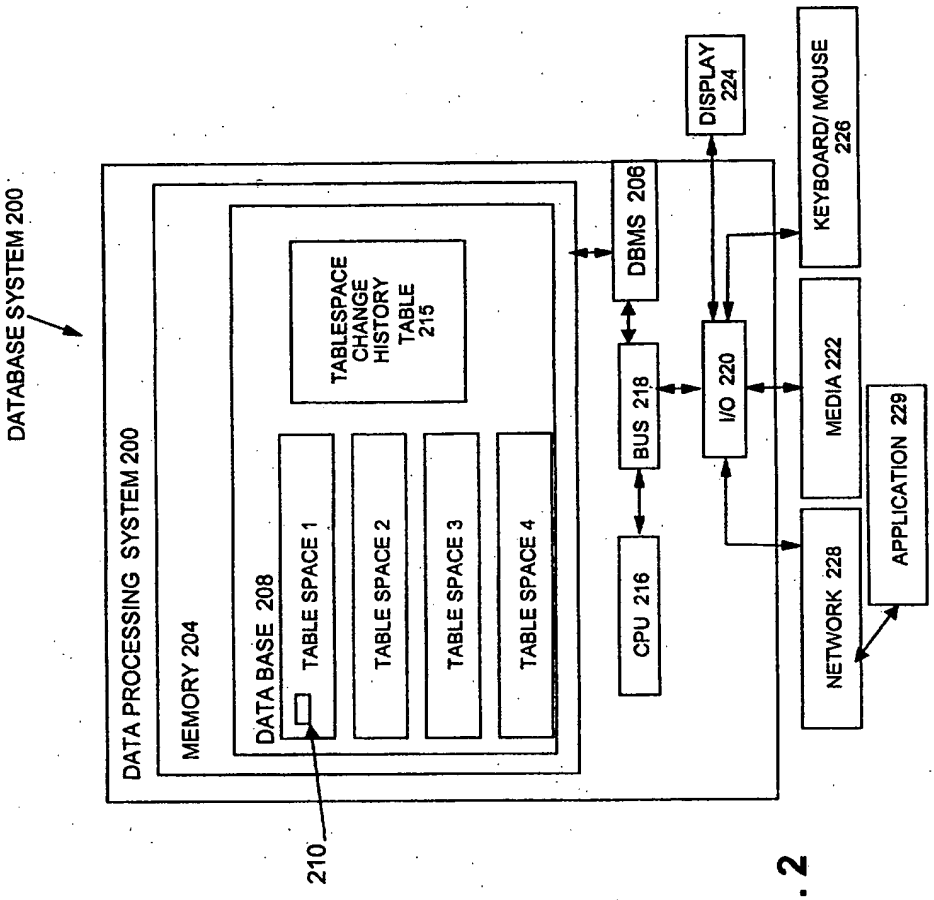

DATABASE SYSTEM 200
DATA PROCESSING SYSTEM 200
MEMORY 204
DATA BASE 208
TABLE SPACE 1
TABLE SPACE 2
TABLE SPACE 3
TABLE SPACE 4
TABLESPACE CHANGE HISTORY TABLE 215
210
DBMS 206
BUS 218
CPU 216
I/O 220
DISPLAY 224
KEYBOARD/ MOUSE 226
MEDIA 222
APPLICATION 229
NETWORK 228

FIG. 2

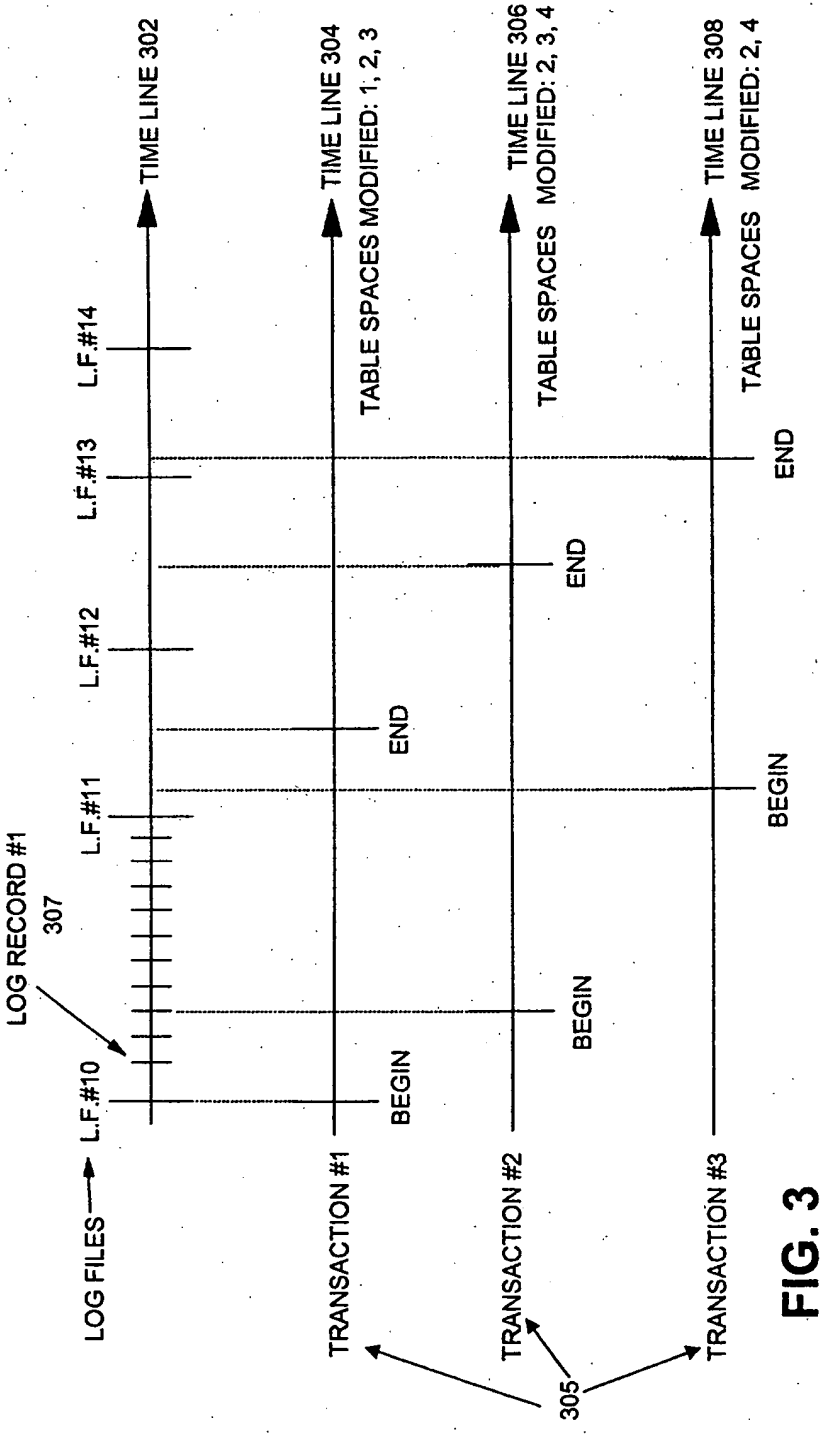

LOG FILES
L.F.#10
L.F.#11
L.F.#12
L.F.#13
L.F.#14
LOG RECORD #1
307
TIME LINE 302
TIME LINE 304
TABLE SPACES MODIFIED: 1, 2, 3
TIME LINE 306
TABLE SPACES MODIFIED: 2, 3, 4
TIME LINE 308
TABLE SPACES MODIFIED: 2, 4
BEGIN
END
BEGIN
END
BEGIN
END
TRANSACTION #1
TRANSACTION #2
TRANSACTION #3
305

FIG. 3

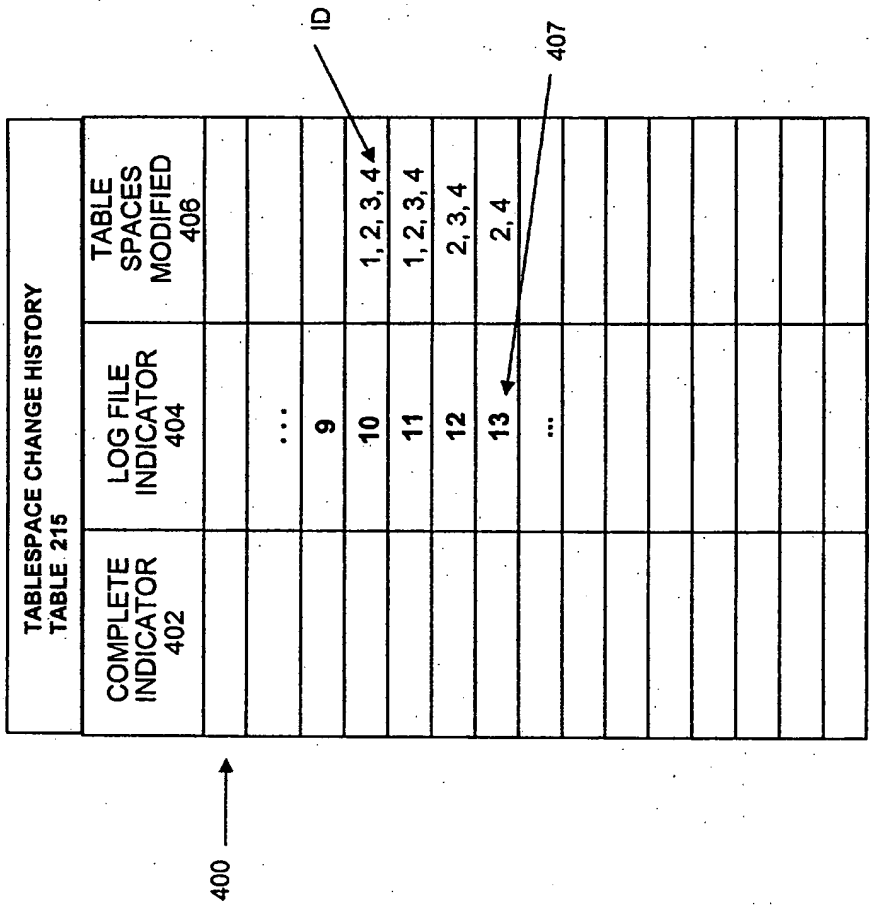

TABLESPACE CHANGE HISTORY TABLE 215
COMPLETE INDICATOR 402
LOG FILE INDICATOR 404
TABLE SPACES MODIFIED 406
ID
407
400
9
10
11
12
13
1, 2, 3, 4
1, 2, 3, 4
2, 3, 4
2, 4

FIG. 4

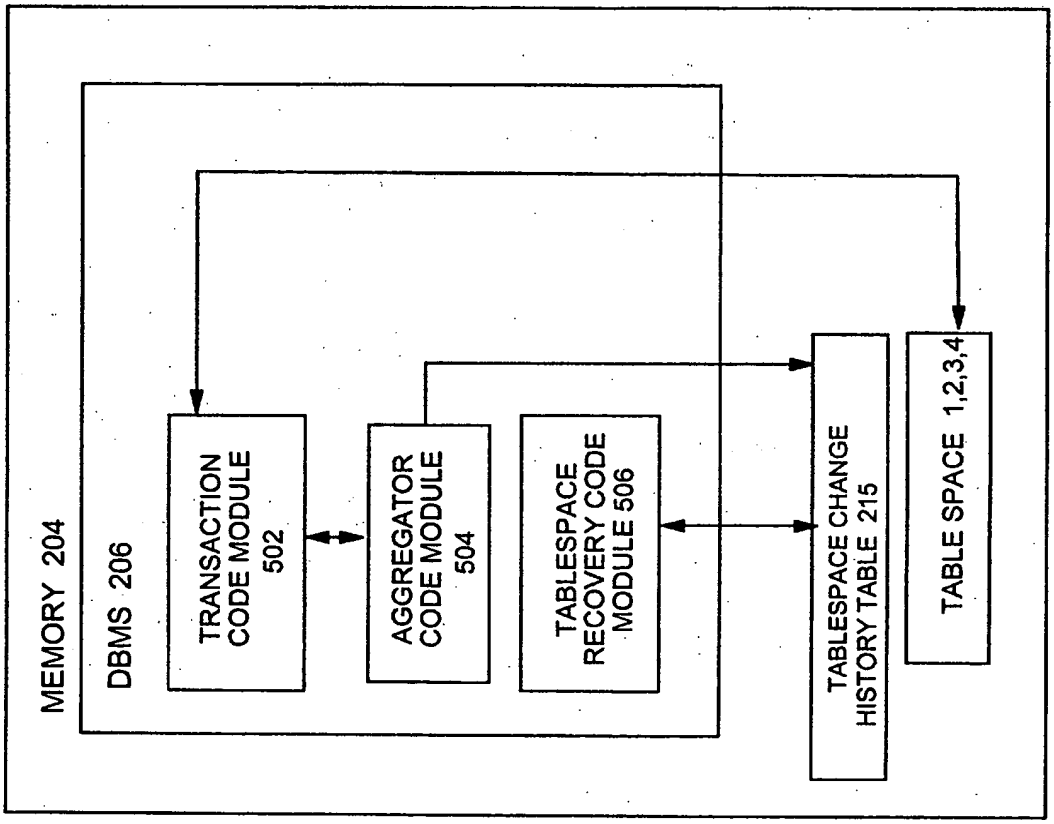

MEMORY 204
DBMS 206
TRANSACTION CODE MODULE 502
AGGREGATOR CODE MODULE 504
TABLESPACE RECOVERY CODE MODULE 506
TABLESPACE CHANGE HISTORY TABLE 215
TABLE SPACE 1,2,3,4

FIG. 5

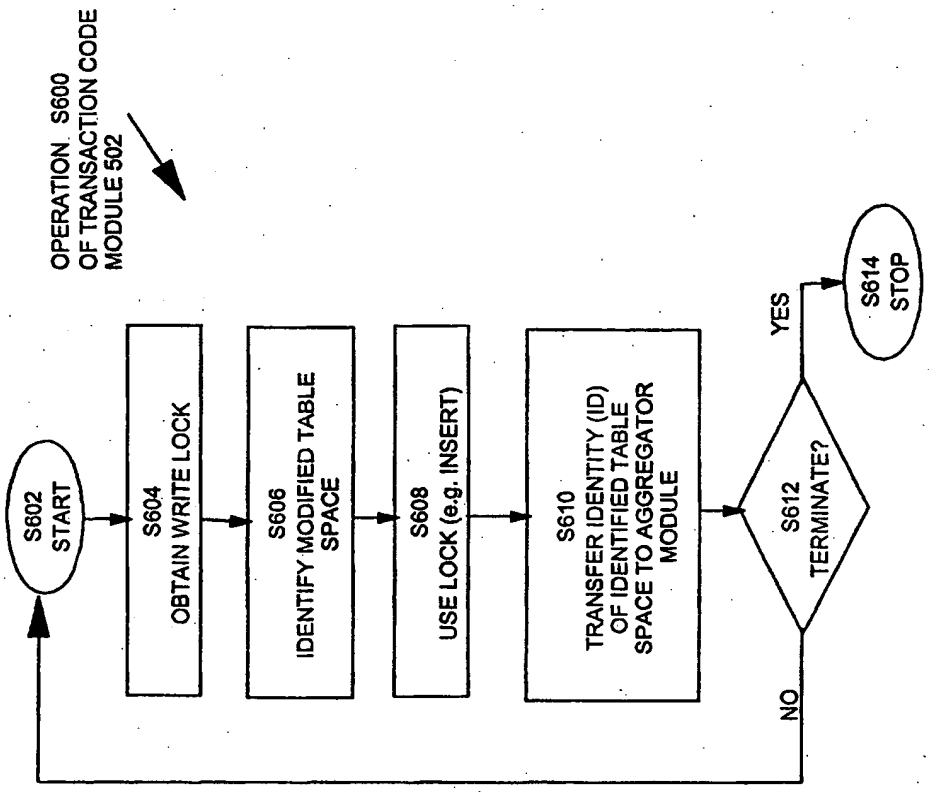

OPERATION S600
OF TRANSACTION CODE
MODULE 502
S602
START
S604
OBTAIN WRITE LOCK
S606
IDENTIFY MODIFIED TABLE
SPACE
S608
USE LOCK (e.g. INSERT)
S610
TRANSFER IDENTITY (ID)
OF IDENTIFIED TABLE
SPACE TO AGGREGATOR
MODULE
S612
TERMINATE?
YES
NO
S614
STOP

FIG. 6

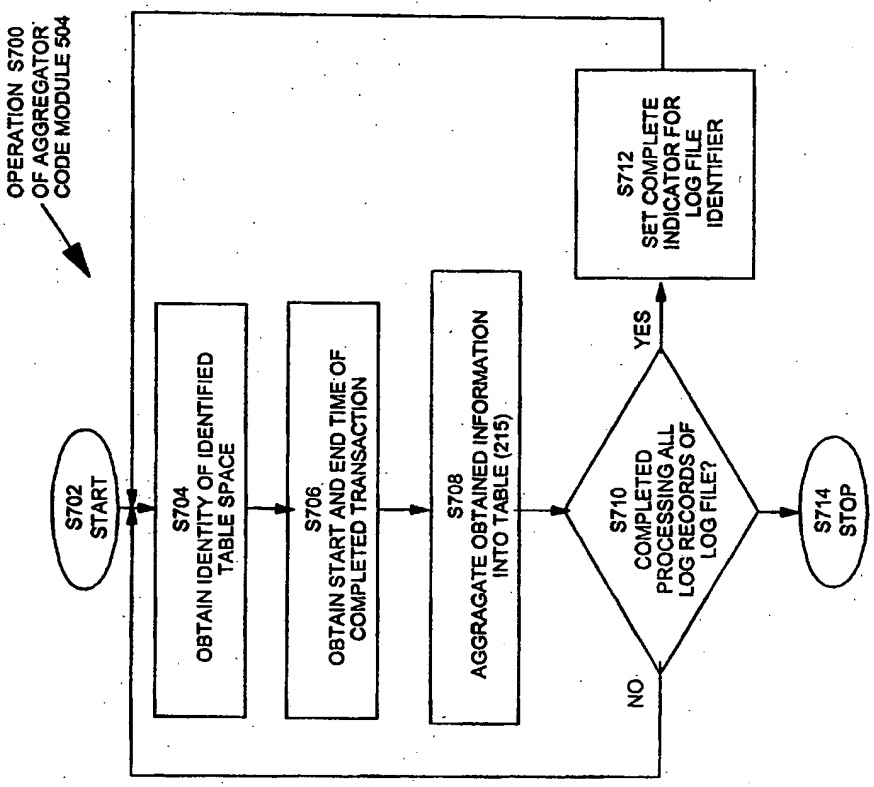
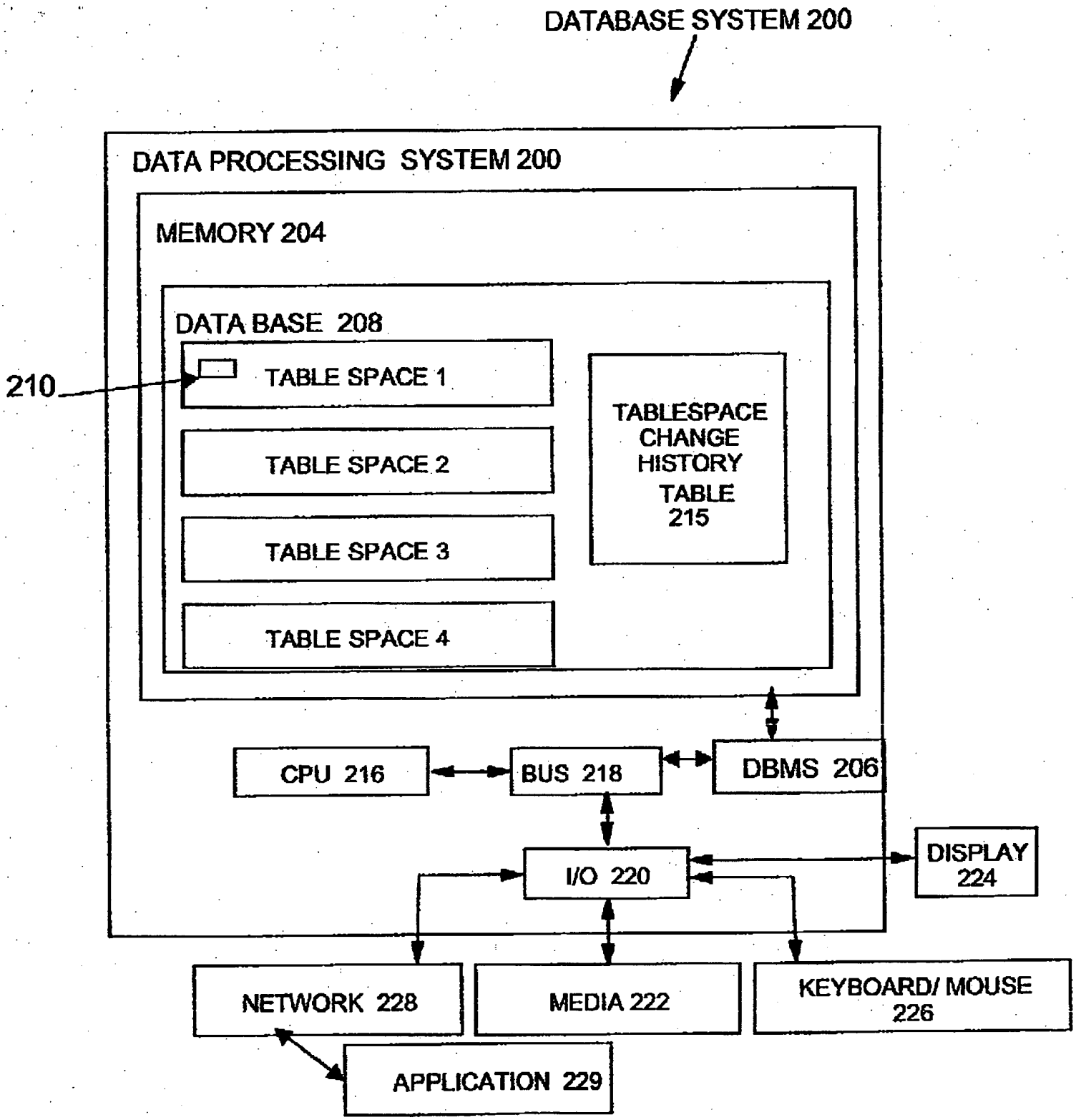

OPERATION S700
OF AGGREGATOR
CODE MODULE 504
S702
START
S704
OBTAIN IDENTITY OF IDENTIFIED
TABLE SPACE
S706
OBTAIN START AND END TIME OF
COMPLETED TRANSACTION
S708
AGGRAGATE OBTAINED INFORMATION
INTO TABLE (215)
S710
COMPLETED
PROCESSING ALL
LOG RECORDS OF
LOG FILE?
YES
NO
S712
SET COMPLETE
INDICATOR FOR
LOG FILE
IDENTIFIER
S714
STOP

FIG. 7


DATABASE SYSTEM 200
DATA PROCESSING SYSTEM 200
MEMORY 204
DATA BASE 208
210
TABLE SPACE 1
TABLE SPACE 2
TABLE SPACE 3
TABLE SPACE 4
TABLESPACE CHANGE HISTORY TABLE 215
CPU 216
BUS 218
DBMS 206
I/O 220
DISPLAY 224
NETWORK 228
MEDIA 222
KEYBOARD/ MOUSE 226
APPLICATION 229

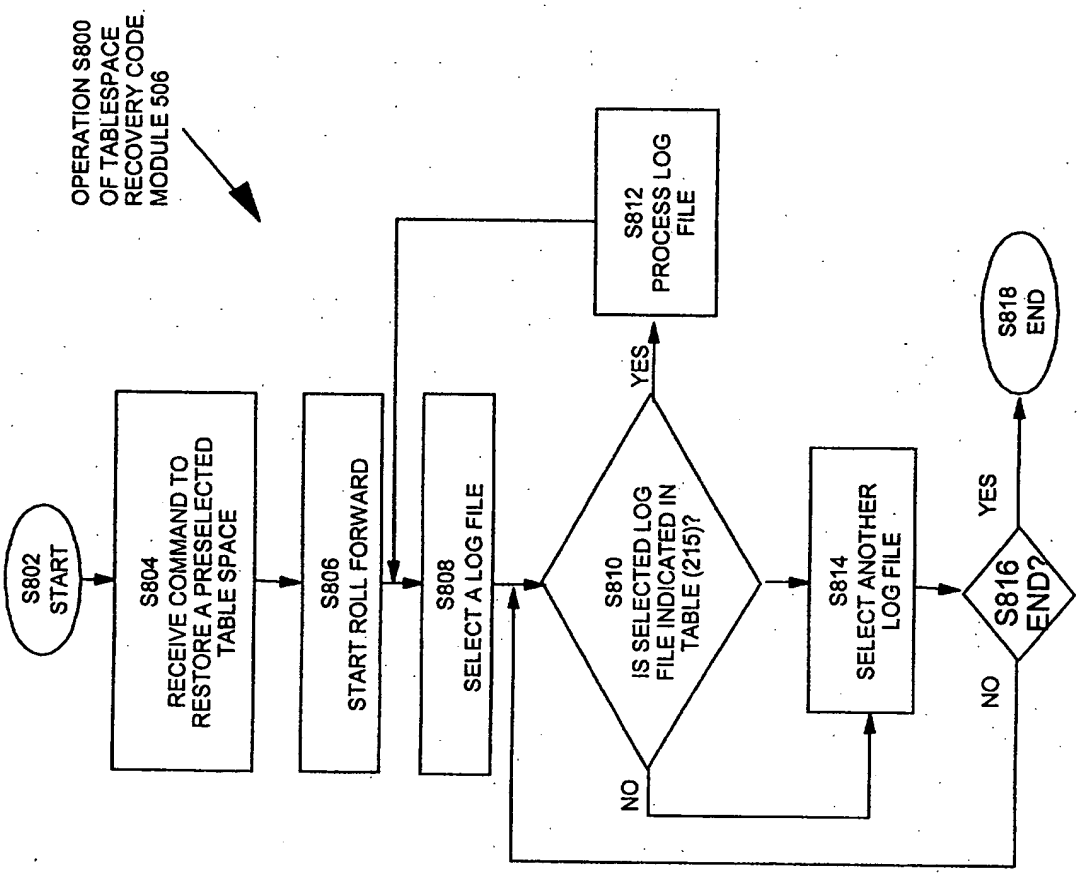

OPERATION S800
OF TABLESPACE
RECOVERY CODE
MODULE 506
S802
START
S804
RECEIVE COMMAND TO
RESTORE A PRESELECTED
TABLE SPACE
S806
START ROLL FORWARD
S808
SELECT A LOG FILE
S810
IS SELECTED LOG
FILE INDICATED IN
TABLE (215)?
YES
NO
S812
PROCESS LOG
FILE
S814
SELECT ANOTHER
LOG FILE
S816
END?
YES
NO
S818
END

FIG. 8